



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|-------------|----------------------|---------------------------------|------------------------|
| 10/714,137 | 11/13/2003 | Roger D. Arnold | J0658.0016 | 9941 |
| 38881 7590 08/06/2007 DICKSTEIN SHAPIRO LLP 1177 AVENUE OF THE AMERICAS 6TH AVENUE NEW YORK, NY 10036-2714 | | | | |
| | | | EXAMINER KAWSAR, ABDULLAH AL | |
| | | | ART UNIT 2109 | PAPER NUMBER |
| | | | MAIL DATE 08/06/2007 | DELIVERY MODE PAPER |

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary

Application No.

10/714,137

Applicant(s)

ARNOLD ET AL.

Examiner

Abdullah-Al Kawsar

Art Unit

2109

**– The MAILING DATE of this communication appears on the cover sheet with the correspondence address –
Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
 - Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 13 November 2003.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 13 November 2003 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
- Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
- Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413) |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) <input checked="" type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date <u>01/09/2007</u> . | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. Claims 1-20 are pending.

Claim Objections

2. Claim 3 is objected to because of the following informalities: "MPV" should spell out. Appropriate correction is required.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Claims 1, 10 and 17 recite the limitation "multiple virtual machine" including a physical processor. There is insufficient antecedent basis for this limitation in the claims.

5. Claim 3 recites "non-volatile memory device comprises a second discretely packaged semiconductor device" which is unclear and does not disclose specifically what is the actual device.

6. Claims 2, 4-9, 11-16 and 18-20 are dependent claims of claim 1, 10 and 17. Therefore they are rejected under the same rationale.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

8. Claims 1, 2, 4, 7-11, 13, 16, 17, and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over “Processor Scheduling on Multiprogrammed, Distributed Memory Parallel Computers” by Sanjeev K. Setia(Setia) in view of Ferrell et al. US Patent No. 5,630,128.

As per claim 1, Setia discloses:

- a method for operating a multiple virtual processor system, the multiple virtual processor system including a program memory, a thread scheduling mechanism, and a physical processor, the method comprising (page 1 col 2 lines 25-30 “When a program is scheduled on a multicomputer, a single kernel process or virtual processor is typically created on each processor allocated to the program. The program’s work is divided among these virtual processors either statically, i.e., at program creation time, or dynamically, i.e., during program execution”) a multi-computer system includes a physical processor inherently.

- *suspending execution of the first thread when the YIELD instruction is processed by the physical processor* (page 3 col 1 lines 25-26, “When a process blocks on a read, it is rescheduled and placed in a blocked queue”)

- *identifying a second thread from the plurality of threads for execution by the physical processor, wherein the second thread is selected by the thread scheduling mechanism based on a predefined schedule and the processed YIELD instruction* (page 3 col 1 lines 25-28, “When a process blocks on a read, it is rescheduled and placed in a blocked queue, and the processor context switches to the next process in the ready queue”) when a process is blocked(yield) it switches to the second thread(next thread) in the ready queue(predefined schedule).

- *executing the second thread by systematically passing second instructions associated with the second thread from the program memory to the physical processor, and causing the physical processor to process the second instructions.* (page 3 col 1 lines 25-28, “When a process blocks on a read, it is rescheduled and placed in a blocked queue, and the processor context switches to the next process in the ready queue”) context switching to the next process means executing the second instruction on the processor.

However Setia does not specifically disclose, *threads comprises a plurality of first instructions including a YIELD instruction.*

Art Unit: 2109

On the other hand Ferrell discloses:

- storing a plurality of threads in the program memory, wherein a first thread of the plurality of threads comprises a plurality of first instructions including a YIELD instruction (col 1 lines 43-47, “operating system 10 is programmed according to the present invention to schedule execution of application program threads constituting one or more of the application programs” and col 10 lines 53-56, “Operating system 10 also includes a ThreadYield function which allows a currently executing program thread to relinquish control of the CPU without blocking or suspending itself or otherwise becoming non-dispatchable” and col 11 lines 1-4, “An application program developer may code a call to the ThreadYield function at any point in the program thread where another thread should execute”)

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Ferrell into method of Setia to have a YIELD instruction. The modification would have been obvious because one of the ordinary skills of the art would have a YIELD instruction to switch a thread with another thread from execution to have a better thread scheduling and system performance.

- executing the first thread by systematically passing the first instructions from the program memory to the physical processor, and causing the physical processor to process the first instructions (col 1 lines 12-13, “a computer processor executes computer programs or program subroutines serially”) program subroutines includes thread and thread includes instructions that are executed serially(systematically passing the first instruction) on the processor.

As per claim 4, the rejection of claim 1 incorporates and further Setia discloses:

- wherein a portion of the program memory comprises a deterministic memory for continuously storing a pre-selected thread of the plurality of threads, and wherein storing the plurality of threads includes writing all instructions associated with the pre-selected thread into the deterministic memory during a system initialization period (page 4 lines 10-19, “an application forks a kernel-schedulable thread of execution..... removes it from the structure and initiates its execution on its virtual processor”)

As per claim 8, the rejection of claim 1 incorporates and further Setia discloses:

- suspending execution of the second thread based on the predefined schedule; and resuming execution of the first thread (page 3 col 1 lines 28-29, “When a message arrives for a blocked process, it is unblocked and returned to the readv queue”)

As per claim 9, the rejection of claim 1 incorporates and further Setia discloses:

- wherein suspending execution of the first thread further comprises determining whether the second thread is available for execution (page 4 lines 22-25, “Application threads typically run on virtual processors until they either block or terminate. At that time, a scheduler thread regains control and looks for another ready application thread to run.”)

Art Unit: 2109

As per claim 2 Setia discloses all the elements of claim 2 except, *storing the plurality of threads comprises writing the plurality of threads from a non-volatile memory device into the program memory.*

On the other hand Ferrell Discloses:

- *wherein the program memory comprises a volatile memory device, and wherein storing the plurality of threads comprises writing the plurality of threads from a non-volatile memory device into the program memory* (col 1 lines 31-35, “operating system 10 is preferably programmed in executable form onto a computer readable medium such as a magnetic disk or tape, loaded into a computer memory and executed on one or more of the CPUs” and col 1 lines 43-47, “operating system 10 is programmed according to the present invention to schedule execution of application program threads constituting one or more of the application programs”) programs in the operating system stored on magnetic disk(non-volatile memory device) are loaded(writing) in the computer memory(volatile memory device) for execution on CPU.

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Ferrell into method of Setia to write thread from non-volatile memory device to program memory. The modification would have been obvious because one of the ordinary skills of the art would have to copy the program thread from non-volatile device to a program memory for execution.

As per claim 7 Setia discloses all the elements of claim 7 except, *executing the first thread comprises selecting the first thread from the plurality of threads based on the predefined schedule.*

Art Unit: 2109

On the other hand Ferrell Discloses:

- wherein executing the first thread comprises selecting the first thread from the plurality of threads based on the predefined schedule (col 4 lines 3-9, "application programs (as written by application program developers) select each thread's priority and dispatch class based on the following principles of operating system, the highest priority available thread from each dispatch class is queued on a run list for execution, the highest priority thread on the run list is executed first")

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Ferrell into method of Setia to have a predefined schedule. The modification would have been obvious because one of the ordinary skills of the art would have a predefined schedule for execution on the CPU for fast and efficient execution of program threads.

As per claim 10, Setia discloses:

- a program memory for storing a plurality of threads; and a processor core coupled to the program memory, the processor core including: a thread scheduling mechanism for scheduling the execution of a first thread and a second thread based on a predetermined schedule, a physical processor for processing instructions associated with a selected thread of the first and second threads (page 1 col 2 lines 25-30 "When a program is scheduled on a multicomputer, a single kernel process or virtual processor is typically created on each processor allocated to the program. The program's work is divided among these virtual

Art Unit: 2109

processors either statically, i.e., at program creation time, or dynamically, i.e., during program execution”) a multi-computer system includes a physical processor inherently.

- wherein the thread scheduling mechanism includes means for suspending execution of the first thread and for initiating execution of the second thread by the physical processor based on the predefined schedule and the processed YIELD instruction (page 3 col 1 lines 25-28, “When a process blocks on a read, it is rescheduled and placed in a blocked queue, and the processor context switches to the next process in the ready queue”) when a process is blocked(yield) it switches to the second thread(next thread) in the ready queue(predefined schedule)

However Setia does not specifically disclose, *threads comprises a plurality of first instructions including a YIELD instruction.*

On the other hand Ferrell discloses:

- wherein the first thread includes a YIELD machine instruction, wherein the processor core comprises means for notifying the thread scheduling mechanism when the YIELD machine instruction is processed by the physical processor during execution of the first thread (col 1 lines 43-47, “operating system 10 is programmed according to the present invention to schedule execution of application program threads constituting one or more of the application programs” and col 10 lines 53-56, “Operating system 10 also includes a ThreadYield function which allows a currently executing program thread to relinquish control of the CPU without blocking or suspending itself or otherwise becoming non-dispatchable” and

Art Unit: 2109

col 11 lines 1-4, "An application program developer may code a call to the ThreadYield function at any point in the program thread where another thread should execute")

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Ferrell into method of Setia to have a YIELD instruction. The modification would have been obvious because one of the ordinary skills of the art would have a YIELD instruction to switch a thread with another thread from execution to have a better thread scheduling and system performance.

- switching means for passing instructions associated with the selected thread from the program memory to the physical processor (col 1 lines 12-13, "a computer processor executes computer programs or program subroutines serially") program subroutines includes thread and thread includes instructions that are executed serially(systematically passing the first instruction) on the processor.

As per claim 17, Setia discloses:

- means for storing a plurality of threads in the program memory, the plurality of threads including a first thread comprising a plurality of first instructions including a YIELD instruction (page 1 col 2 lines 25-30 "When a program is scheduled on a multicomputer, a single kernel process or virtual processor is typically created on each processor allocated to the program. The program's work is divided among these virtual processors either statically, i.e., at program creation time, or dynamically, i.e., during program execution") a multi-computer system includes a physical processor inherently.

- means for suspending execution of the first thread upon determining that the YIELD instruction has been processed by the physical processor (page 3 col 1 lines 25-26, “When a process blocks on a read, it is rescheduled and placed in a blocked queue”)

- means for identifying and executing a second thread from the plurality of threads using the physical processor, wherein the second thread is selected based on a predefined schedule and the processed YIELD instruction (page 3 col 1 lines 25-28, “When a process blocks on a read, it is rescheduled and placed in a blocked queue, and the processor context switches to the next process in the ready queue”) context switching to the next process means executing the second instruction on the processor.

However Setia does not specifically disclose, *threads comprises a plurality of first instructions including a YIELD instruction.*

On the other hand Ferrell discloses:

- means for executing the first thread by systematically passing the first instructions from the program memory to the physical processor, and causing the physical processor to process the first instructions (col 1 lines 43-47, “operating system 10 is programmed according to the present invention to schedule execution of application program threads constituting one or more of the application programs” and col 10 lines 53-56, “Operating system 10 also includes a ThreadYield function which allows a currently executing program thread to relinquish control of the CPU without blocking or suspending itself or otherwise becoming non-dispatchable” and col

Art Unit: 2109

11 lines 1-4, "An application program developer may code a call to the ThreadYield function at any point in the program thread where another thread should execute")

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Ferrell into method of Setia to have a YIELD instruction. The modification would have been obvious because one of the ordinary skills of the art would have a YIELD instruction to switch a thread with another thread from execution to have a better thread scheduling and system performance.

- means for determining when the YIELD instruction is processed by the physical processor (col 1 lines 12-13, "a computer processor executes computer programs or program subroutines serially") program subroutines includes thread and thread includes instructions that are executed serially(systematically passing the first instruction) on the processor.

Claims 11 and 13 have the same claim limitation of claims 2 and 4 above. They are therefore rejected under the same rational.

Claims 16 and 20 have the same claim limitation of claim 9 above. They are therefore rejected under the same rational.

9. Claims 6, 15 and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Processor Scheduling on Multiprogrammed, Distributed Memory Parallel Computers" by

Art Unit: 2109

Sanjeev K. Setia(Setia) in view of Ferrell et al. US Patent No. 5,630,128 and in view of Propescu et al. US Patent No. 5,487,156.

As per claim 6 the combined method of Setia and Ferrell discloses all the elements of claim 6 except, *first thread comprises fetching the first instructions from the program memory using a first program counter, and wherein executing the second thread comprises fetching the second instructions from the program memory using a second program counter.*

However Propescu discloses:

- *wherein executing the first thread comprises fetching the first instructions from the program memory using a first program counter, and wherein executing the second thread comprises fetching the second instructions from the program memory using a second program counter* (col 1 lines 22-35, "In the first stage, an instruction is fetched from memory at a location identified by a program counter which points to the latest fetched instruction,.....

Another program counter, the update-virtual PC, identifies the instruction that just completed updating the state of the processor.")

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Propescu into the combined method of Setia and Ferrell to have program counter to fetch instructions. The modification would have been obvious because one of the ordinary skills of the art would use program counter to fetch instructions for better identification and execution of instructions.

Claims 15 and 19 have the same claim limitation of claim 6 above. They are therefore rejected under the same rational.

10. Claims 3, 5, 12, 14 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Processor Scheduling on Multiprogrammed, Distributed Memory Parallel Computers" by Sanjeev K. Setia(Setia) in view of Ferrell et al. US Patent No. 5,630,128 and in view of Jagannathan et al. US Patent No. 5,692,193.

As per claim 3 the combined method of Setia and Ferrell discloses all the elements of claim 3 except, *first discretely packaged semiconductor device, and the non-volatile memory device comprises a second discretely packaged semiconductor device, and wherein writing the plurality of threads comprises transmitting data between the first and second discretely packaged semiconductor devices.*

However Jagannathan discloses:

- wherein the MVP system comprises a first discretely packaged semiconductor device, and the non-volatile memory device comprises a second discretely packaged semiconductor device, and wherein writing the plurality of threads comprises transmitting data between the first and second discretely packaged semiconductor devices during operation of the MVP system (page 1 abstract lines 6-13, "A virtual machine comprises a virtual address space and a set of virtual processors that are connected in a virtual topology. Virtual machines are mapped onto abstract physical machines with each virtual processor mapped onto an abstract physical processor. The third layer of abstraction defines threads. Threads are lightweight processes that

Art Unit: 2109

run on virtual processors.”) semiconductor devices are processor and memory of a computer and threads being executed on virtual processor(transferring data between then).

Therefore, it would have been obvious to a person of ordinary skill in art at the time of invention was made to incorporate the teaching of Jagannathan into the combined method of Setia and Ferrell to transfer data between semiconductor device during execution. The modification would have been obvious because one of the ordinary skills of the art would have to transfer threads from memory to processor for execution in a virtual processing system.

As per claim 5, the rejection of claim 1 incorporates and further Jagannathan discloses:

- wherein the first thread includes operating state information that is loaded into the physical processor before executing the first thread (col 5 lines 2-4, “Virtual processors execute on physical processors which are abstractions of actual physical computing devices” and col 5 lines 57-59, “A virtual processor is an abstraction that defines scheduling, migration and load balancing policies for the threads it executes.”) virtual processor includes the thread information which is executed on the physical processor.

Claims 14 and 18 have the same claim limitation of claim 5 above. They are therefore rejected under the same rational.

Claim 12 have the same claim limitation of claim 3 above. It is therefore rejected under the same rational.

Conclusion

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

TITLE: Controlled scheduling of program threads in a multitasking operating system, US Patent No. 5,630,128.

TITLE: Software architecture for control of highly parallel computer systems, US Patent No. 5,692,193.

TITLE: Processor architecture having independently fetching issuing and updating operations of instructions which are sequentially assigned and stored in order fetched, US Patent No. 5,487,156.

TITLE: Processor Scheduling on Multiprogrammed, Distributed Memory Parallel Computers; Sanjeev K. Setia, Mark S. Squillante, Satish K. Tripathi ; Proceedings of the 1993 ACM SIGMETRICS conference on Measurement and modeling of computer systems.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Abdullah-Al Kawsar whose telephone number is 571-270-3169.

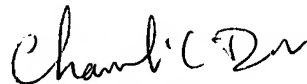
The examiner can normally be reached on 7:30am to 5:00pm, EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Chameli Das can be reached on 571-272-3696. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Art Unit: 2109

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

AK



CHAMELI DAS
SUPERVISORY PATENT EXAMINER

8/1/07